

REMARKS

Claims 1-3, 8, 11-14, and 19-32 are pending. Claims 4-7, 9, 10, and 15-18 are canceled. Claims 1-3, 8, and 29-31 are amended. Claim 32 is added. Support for the amendments and new claim 32 may be found at least in originally filed claims 4-7, 9 and 10, as well as the originally filed specification on page 8, line 24, to page 16, line 25. Applicants respectfully request reconsideration of the claims in view of the amendments and the following remarks.

I. Examiner Interview

Applicants thank the Examiner for the courtesies extended during the telephone interview on April 18, 2006. Applicants presented proposed amendments and argued that the amendments overcome the current set of rejections. Applicants also argued that claim 19 overcomes the applied prior art. Examiner agreed to reconsider the rejections in view of the amendments and the remarks presented below.

II. 35 U.S.C. § 102, Alleged Anticipation of Claims 1-3 and 29-31

The Office rejects claims 1-3 and 29-31 under 35 U.S.C. § 102(b) as allegedly being anticipated by *Cherazi et al.* (U.S. Patent No. 6,282,556). This rejection is respectfully traversed.

Cherazi teaches a high performance pipelined data path for a media processor. The pipelined data path architecture of *Cherazi* requires two execution pipestages to perform all instructions including wide data format multiply instructions. As one example, the architecture of *Cherazi* performs a sum of absolute differences (SABD) instruction; however, *Cherazi* is primarily concerned with performing multiply operations. See *Cherazi*, Abstract.

The Office alleges that *Cherazi* teaches a byte execution unit. However, it is clear that *Cherazi* is concerned with an overall architecture for performing multiply operations, rather than a specific byte execution unit. In fact, the term "byte execution" does not appear anywhere in *Cherazi*. While *Cherazi* does teach that the multiply circuit is

capable of 8x8 multiply operations, *Cherazi* does not teach the byte execution unit recited in claim 1, for example.

Claim 1 is amended to incorporate limitations originally presented in claims 4-7, now canceled. That is, claim 1 recites a byte execution unit that may receive a count ones in bytes operation, an average bytes operation, an absolute differences of bytes operation, or a sum bytes into halfwords operation. Responsive to a count ones in bytes operation, the byte execution unit counts a number of logical one bits in each byte of at least a first operand and stores each result in a corresponding byte of a destination register. Responsive to an average bytes operation, the byte execution unit averages each byte of the first operand with a corresponding byte of a second operand and stores each result in a corresponding byte of the destination register. Responsive to an absolute difference of bytes operation, the byte execution unit subtracts each byte of the first operand from a corresponding byte of the second operand to form an intermediate result, determines an absolute value of each intermediate result to form a final result, and stores each final result in a corresponding byte of the destination register. Responsive to a sum bytes into halfwords operation, the byte execution unit sums a number of corresponding one-byte portions of the first operand or the second operand and stores each result in a corresponding halfword of the destination register. *Chehrizi* does not teach a byte execution unit that is configured to perform a count ones in bytes operation, an average bytes operation, an absolute difference of bytes operation, and a sum bytes into halfwords operation within the same byte execution unit.

With respect to claim 4, the Office Action acknowledges that *Cherazi* does not teach byte instruction information that specifies a count ones in byte operation. Rather, the Office Action relies on *Peleg et al.* (U.S. Patent No. 6,070,237) as allegedly teaching this feature. The Office Action then concludes that it would have been obvious to include the population count operation of *Peleg* in the pipeline data path architecture of *Chehrizi*. The motivation proposed in the Office Action is that *Peleg* teaches that doing so improves the performance of algorithms requiring the totaling of the number of bits set. Applicants respectfully disagree. *Peleg* does not teach "doing so," because *Peleg* does not provide any suggestion whatsoever that the population count operation of *Peleg* could be combined with the high performance pipelined data path architecture of *Chehrizi*.

Therefore, *Peleg* could not possibly suggest the specific combination proposed in the Office Action. Obviousness cannot be established by combining the teachings of the prior art to produce the claimed invention absent some teaching, suggestion, or incentive supporting the combination. See *In re Geiger*, 815 F.2d 686, 688, 2 U.S.P.Q.2d 1276, 1278 (Fed. Cir. 1987).

Moreover, the Office may not use the claimed invention as an "instruction manual" or "template" to piece together the teachings of the prior art so that the invention is allegedly rendered obvious. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). Such reliance is an impermissible use of hindsight with the benefit of Applicants' disclosure. *Id.* Therefore, absent some teaching, suggestion, or incentive in the prior art, *Chehrizi* and *Peleg* cannot be properly combined to form the claimed invention. As a result, absent any teaching, suggestion, or incentive from the prior art to make the proposed combination, the presently claimed invention can be reached only through impermissible use of hindsight with the benefit of Applicants' disclosure a model for the needed changes.

Furthermore, the Office Action simply presents the combination as obvious without any explanation of how a person of ordinary skill in the art could combine the population count operation of *Peleg* with the pipelined data path architecture for wide data format operations taught by *Chehrizi*. The Office Action proffers no explanation of how such a combination would somehow form an operational device. Combining functions from complex logic circuits is no easy task. Therefore, there must be some teaching in the prior art enabling the combination, or establishing a reasonable likelihood of success, in order for the combination to be proper.

With respect to claim 7, the Office Action acknowledges that *Chehrizi* does not teach or suggest byte instruction information that specifies a sum bytes into halfwords operation. Rather, the Office Action relies on *Oberman et al.* (AMD 3DNow! Technology: Architecture and Implementations) as allegedly teaching this feature as a "PFACC instruction." Applicants respectfully disagree. *Oberman* does indeed appear to teach a PFACC instruction, which is a packed floating-point accumulation operation that operates on 32-bit operands. Clearly, this operation is not a sum bytes into halfwords operation that sums a number of corresponding one-byte portions of a first operand or

the second operand and stores each result in a corresponding halfword of the destination register.

The Office Action concludes that it would have been obvious to a person of ordinary skill in the art to combine the floating-point accumulation operation of *Oberman* with the high performance pipelined data path architecture of *Chehrizi*. However, the Office Action simply presents the combination as obvious without any explanation of how a person of ordinary skill in the art could combine the floating-point accumulation operation of *Oberman* with the pipelined data path architecture for wide data format operations taught by *Chehrizi*. The Office Action proffers no explanation of how such a combination would somehow form the byte execution unit of claim 1. Combining functions from complex logic circuits is no easy task. Therefore, there must be some teaching in the prior art enabling the combination, or establishing a reasonable likelihood of success, in order for the combination to be proper.

Still further, claim 1, for example, recites a byte execution unit that is configured to perform four byte operations, namely a count ones in bytes operation, an average bytes operation, an absolute difference of bytes operation, and a sum bytes into halfwords operation within the same byte execution unit. The Office Action has not presented a reference that teaches a single byte execution unit capable of performing these four operations. To the contrary, even assuming, *arguendo*, that the references teach what is alleged in the Office Action, the applied references present three different architectures for performing the various operations. The mere presence of these three different architectures is evidence that a single byte execution unit for performing the four different byte operations would not have been obvious to a person of ordinary skill in the art.

Independent claim 29 recites subject matter addressed above with respect to claim 1 and is allowable for similar reasons. Because dependent claims 2, 3, 30, and 31 depend from claims 1 and 29, the same distinctions between claims 1 and 29 and the applied prior art apply for these claims. In addition, claims 2, 3, 30, and 31 recite further limitations not taught or suggested by the prior art.

Therefore, Applicants respectfully request withdrawal of the rejection of claims 1-3 and 29-31 under 35 U.S.C. § 102.

III. 35 U.S.C. § 103, Alleged Obviousness of Claims 8, 12, and 13

The Office rejects claims 8, 12, and 13 under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Oberman et al.* (AMD 3DNow! Technology: Architecture and Implementations). This rejection is respectfully traversed.

Oberman teaches the AMD 3DNow! architecture for the AMD-K6-2 microprocessor. Oberman teaches twenty-one 3DNow! instructions, which include sixteen packed floating point (32-bit) operations, one packed integer double (32-bit) operation, one packed 16-bit integer operation, one packed 8-bit unsigned integer average, one fast entry/exit of MMX operation, and one data prefetch operation. Thus, Oberman teaches only one byte operation, which is the unsigned integer average operation.

In contradistinction, claim 8 recites a byte execution unit comprising a control unit coupled to pre-processing logic, adder logic, and post-processing logic, wherein the byte execution unit may receive a count ones in bytes operation, an average bytes operation, an absolute differences of bytes operation, or a sum bytes into halfwords operation. Responsive to a count ones in bytes operation, the control unit sets control signals to configure the pre-processing logic, the adder logic, and the post-processing logic to count a number of logical one bits in each byte of at least the first operand and to store each result in a corresponding byte of the destination register. Responsive to an average bytes operation, the control unit sets control signals to configure the pre-processing logic, the adder logic, and the post-processing logic to average each byte of the first operand with a corresponding byte of the second operand and to store each result in a corresponding byte of the destination register. Responsive to an absolute differences of bytes operation, the control unit sets control signals to configure the pre-processing logic, the adder logic, and the post-processing logic to subtract each byte of the first operand from a corresponding byte of the second operand to form an intermediate result, to determine an absolute value of each intermediate result to form a final result, and to store each final result in a corresponding byte of the destination register. Responsive to a sum bytes into halfwords operation, the control unit sets control signals to configure the pre-processing logic, the adder logic, and the post-processing logic to sum a number of corresponding one-byte portions of the first operand or the second operand and to store

each result in a corresponding halfword of the destination register. *Oberman* does not teach a byte execution unit that is configured to perform a count ones in bytes operation, an average bytes operation, an absolute difference of bytes operation, and a sum bytes into halfwords operation within the same byte execution unit.

The applied reference fails to teach each and every claim limitation. Therefore, claim 8 is not rendered obvious by the teachings of *Oberman*. Because claims 12 and 13 depend from claim 8, the same distinctions between claim 8 and *Oberman* apply for these claims. In addition, claims 12 and 13 recite further combinations of features not taught or suggested by the applied reference.

Therefore, Applicants respectfully request withdrawal of the rejection of claims 8, 12, and 13 under 35 U.S.C. § 103(a).

III(A). 35 U.S.C. § 103, Alleged Obviousness of Claim 11

The Office rejects claim 11 under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Oberman* in view of *Peleg*. This rejection is respectfully traversed.

Applicants submit that claim 11 is allowable at least by virtue of its dependency on claim 8. *Peleg* does not cure the deficiencies of *Oberman*; therefore, claim 11 is not rendered obvious by the proposed combination of *Oberman* and *Peleg*.

The Office Action acknowledges that *Oberman* does not teach that the pre-processing logic comprises population counter logic coupled to receive operands and configured to produce population output signals indicative of numbers of logical ones in portions of the operands. The Office Action relies on *Peleg* as allegedly teaching this feature and concludes that it would have been obvious to a person of ordinary skill in the art to include the population counter logic of *Peleg* in the 3DNow! architecture of *Oberman*. However, the Office Action simply presents the combination as obvious without any explanation of how a person of ordinary skill in the art could combine the population counter operation of *Peleg* with the 3DNow! architecture taught by *Oberman*. The Office Action proffers no explanation of how such a combination would somehow form the byte execution unit of claim 11. Combining functions from complex logic circuits is no easy task. Therefore, there must be some teaching in the prior art enabling the combination, or establishing a reasonable likelihood of success, in order for the combination to be proper.

Still further, claim 11, for example, recites a byte execution unit that is configured to perform four byte operations, namely a count ones in bytes operation, an average bytes operation, an absolute difference of bytes operation, and a sum bytes into halfwords operation within the same byte execution unit. The Office Action has not presented a reference that teaches a single byte execution unit capable of performing these four operations.

Therefore, Applicants respectfully request withdrawal of the rejection of claim 11 under 35 U.S.C. § 103(a).

III(B). 35 U.S.C. § 103, Alleged Obviousness of Claim 14

The Office rejects claim 11 under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Oberman* in view of *Intel IA-64 Application Developer's Guide*, hereinafter "*IA-64 Guide*." This rejection is respectfully traversed.

Applicants submit that claim 14 is allowable at least by virtue of its dependency on claim 8. *IA-64 Guide* does not cure the deficiencies of *Oberman*; therefore, claim 14 is not rendered obvious by the proposed combination of *Oberman* and *IA-64 Guide*. The Office Action acknowledges that *Oberman* does not teach that the post-processing logic comprises bit shift operations. The Office Action relies on *IA-64 Guide* as allegedly teaching this feature. Again, the Office Action fails to proffer any explanation as to how a feature from the Intel IA-64 Guide can somehow be incorporated into the AMD 3DNow! architecture to result in an operational device, let alone the specific byte execution unit recited in claim 14. Applicants submit that a person of ordinary skill in the art would not be motivated to combine the AMD architecture with the Intel teachings. Furthermore, such a combination would not result in the invention recited in claim 14.

Therefore, Applicants respectfully request withdrawal of the rejection of claim 14 under 35 U.S.C. § 103(a).

IV. 35 U.S.C. § 103, Alleged Obviousness of Claims 19-24 and 28

The Office rejects claims 19-24 and 28 under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Cherazi* in view of *Peleg*. This rejection is respectfully traversed.

The Office alleges that *Cherazi* teaches a byte execution unit comprising a plurality of byte units. Applicants respectfully disagree. *Cherazi* appears to teach a pipelined data path. Within two pipestages, a multiply circuit can perform multiply operations. The data path architecture also may be configured to perform specially adapted multimedia instructions. However, *Cherazi* does not teach a byte execution unit comprising a plurality of byte units. In fact, the terms "byte execution unit" and "byte unit" do not appear anywhere in the reference. The Office Action proffers no explanation as to how the pipeline data path of *Cherazi* is somehow equivalent to a plurality of byte units, particularly the specific byte units recited in claim 19.

Furthermore, the Office alleges that *Cherazi* teaches that each alleged "byte unit" comprises a first compressor unit, a second compressor unit, an adder input multiplexer logic, and an adder logic. Applicants respectfully disagree. As described above, *Cherazi* does not teach a plurality of byte units where each byte unit comprises a first compressor unit and a second compressor unit. Furthermore, *Cherazi* does not teach that the first compressor unit, component 326a, is coupled to receive a portion of the first operand and configured to produce a first plurality of compressor output signals dependent upon the first operand. *Cherazi* also fails to teach that the second compressor unit, component 326b, is coupled to receive a portion of the second operand and configured to produce a second plurality of compressor output signals dependent upon the second operand. The Office Action alleges *Cherazi* teaches this feature at col. 7, line 56, to col. 8, line 9; col. 10, lines 2-6. *Cherazi* states:

The partitioned multiplier circuit 314 is described in co-pending United States patent application entitled, "A High Performance Universal Multiplier Circuit," filed on Oct. 8, 1999, Ser. No. 09/415,485 by Chehrazzi, Oklobdzija and Farooqui, attorney docket SONY-50N3285. The multiplier circuit 314 consists of four 32x32 partitioned multipliers. Each 32x32 partitioned multiplier of circuit 314 can be partitioned to operate in the modes as shown in FIG. 6A, FIG. 6B and FIG. 6C which are described below. Booth encoding is used to generate 17 partial products from the first and second 128-bit operands. Circuitry for performing Booth encoding is described in co-pending United States patent application, entitled "Multiplier Circuit Having Optimized Booth Encoder/Selector," filed on Mar. 29, 1999, Ser. No. 09/280,176 by Chehrazzi, Oklobdzija and

Farooqui. These 17 partial products are compressed using a compressor tree into one 256-bit sum output (stored in pipeline register 316) which is supplied over 256-bit bus 356 and one 256-bit carry output (stored in pipeline register 318) which is supplied over 256-bit bus 358. These are the compressed partial products. Depending on the partition mode employed, the compressed partial products stored in registers 316 and 318 can represent one or multiple results. Multiply circuit 314 processes the multiply instructions MUL, MULD and MADD.

Cherazi, col. 7, line 56, to col. 8, line 9. This portion of *Cherazi* teaches that Booth encoding is used to generate seventeen partial products and the first compressor unit is used to compress these partial products. The cited portion of *Cherazi* does not teach that the first compressor unit is part of one of a plurality of byte units or that the first compressor unit is coupled to receive a portion of the first operand.

Cherazi also states:

The logic unit includes two separate eight 8-bit subtractors 324b and 322b, an 8-input 8-bit 4:2 compressor 326b, a universal 64-bit adder 320b, a logic operations circuit 328b and a multiplexer circuit 332b.

Cherazi, col. 10, lines 27-31. This portion of *Cherazi* teaches that the logic unit of FIG. 5B includes an 8-input 8-bit 4:2 compressor. However, *Cherazi* does not teach that the second compressor unit is part of one of a plurality of byte units or that the second compressor unit is coupled to receive a portion of the second operand.

Still further, the Office Action alleges that *Cherazi* teaches adder input multiplexer logic and adder logic. However, the Office Action provides no analysis as to why these components are interpreted to be equivalent to a byte execution unit comprising a plurality of byte units, as recited in claim 19. In fact, the adder logic components cited by the Office Action, components 340a and 340b of *Cherazi*, are actually a 64-bit carry propagate adder circuits that are used in partitioned multipliers. See *Cherazi*, col. 9, line 45-67; col. 10, lines 26-35. Clearly, components 340a and 340b of *Cherazi* cannot be interpreted to be adder logic that is part of one of a plurality of byte units in a byte execution unit.

Further, with respect to claim 19, the Office Action acknowledges that *Cherazi* does not teach a plurality of population counters, each coupled to receive a portion of a

first operand and configured to produce a population output signal indicative of a number of logic ones in the corresponding portion of the first operand and, further, that the output signal is received by the adder input multiplexer. Rather, the Office Action relies on *Peleg* as allegedly teaching this feature. The Office Action then concludes that it would have been obvious to include the population count operation of *Peleg* in the pipeline data path architecture of *Chehrizi*. The motivation proposed in the Office Action is that *Peleg* teaches that doing so improves the performance of algorithms requiring the totaling of the number of bits set. Applicants respectfully disagree. *Peleg* does not teach "doing so," because *Peleg* does not provide any suggestion whatsoever that the population count operation of *Peleg* could be combined with the high performance pipelined data path architecture of *Chehrizi*. Therefore, *Peleg* could not possibly suggest the specific combination proposed in the Office Action. Obviousness cannot be established by combining the teachings of the prior art to produce the claimed invention absent some teaching, suggestion, or incentive supporting the combination. See *In re Geiger*, 815 F.2d 686, 688, 2 U.S.P.Q.2d 1276, 1278 (Fed. Cir. 1987).

Moreover, the Office may not use the claimed invention as an "instruction manual" or "template" to piece together the teachings of the prior art so that the invention is allegedly rendered obvious. *In re Fritch*, 972 F.2d 1260, 23 U.S.P.Q.2d 1780 (Fed. Cir. 1992). Such reliance is an impermissible use of hindsight with the benefit of Applicants' disclosure. *Id.* Therefore, absent some teaching, suggestion, or incentive in the prior art, *Chehrizi* and *Peleg* cannot be properly combined to form the claimed invention. As a result, absent any teaching, suggestion, or incentive from the prior art to make the proposed combination, the presently claimed invention can be reached only through impermissible use of hindsight with the benefit of Applicants' disclosure a model for the needed changes.

Furthermore, the Office Action simply presents the combination as obvious without any explanation of how a person of ordinary skill in the art could combine the population count operation of *Peleg* with the pipelined data path architecture for wide data format operations taught by *Chehrizi*. The Office Action proffers no explanation of how such a combination would somehow form an operational device. Combining functions from complex logic circuits is no easy task. Therefore, there must be some

teaching in the prior art enabling the combination, or establishing a reasonable likelihood of success, in order for the combination to be proper.

Because claims 20-24 and 28 depend from claim 19, the same distinctions between claim 19 and *Chehrazi* and *Peleg* apply for these claims. In addition, claims 20-24 and 28 recite further combinations of features not taught or suggested by the applied prior art. Furthermore, new claim 32, which is dependent on claim 19, is allowable for at least the reasons mentioned above with respect to claim 1.

Therefore, Applicants respectfully request withdrawal of the rejection of claims 19-24 and 28 under 35 U.S.C. § 103(a).

IV(A). 35 U.S.C. § 103, Alleged Obviousness of Claim 25

The Office rejects claim 25 under 35 U.S.C. § 103(a) as allegedly being unpatentable over *Chehrazi* in view of *Peleg*, as applied to claim 19, and further in view of *Oberman*. This rejection is respectfully traversed.

Applicants submit that claim 25 is allowable at least by virtue of its dependency on claim 19. *Oberman* does not cure the deficiencies of *Chehrazi* and *Peleg*; therefore, claim 25 is not rendered obvious by the proposed combination of *Chehrazi*, *Peleg*, and *Oberman*.

The Office Action acknowledges that *Chehrazi* and *Peleg* do not teach a sum bytes into halfwords operation. The Office Action relies on *Oberman* as allegedly teaching this feature as a PFACC instruction. Applicants respectfully disagree. *Oberman* does indeed appear to teach a PFACC instruction, which is a packed floating-point accumulation operation that operates on 32-bit operands. Clearly, this operation is not a sum bytes into halfwords operation that sums a number of corresponding bytes of an operand.

The Office Action concludes that it would have been obvious to a person of ordinary skill in the art to combine the floating-point accumulation operation of *Oberman* with the high performance pipelined data path architecture of *Chehrazi* and the population count on packed data types operation of *Peleg*. However, the Office Action simply presents the combination as obvious without any explanation of how a person of ordinary skill in the art could possibly combine the floating-point accumulation operation

of *Oberman* with the pipelined data path architecture for wide data format operations taught by *Chehrazi* and the population count operation of *Peleg*. The Office Action proffers no explanation of how such a combination would somehow form the byte execution unit of claim 25. Combining functions from complex logic circuits is no easy task. Therefore, there must be some teaching in the prior art enabling the combination, or establishing a reasonable likelihood of success, in order for the combination to be proper.

Therefore, Applicants respectfully request withdrawal of the rejection of claim 25 under 35 U.S.C. § 103(a).

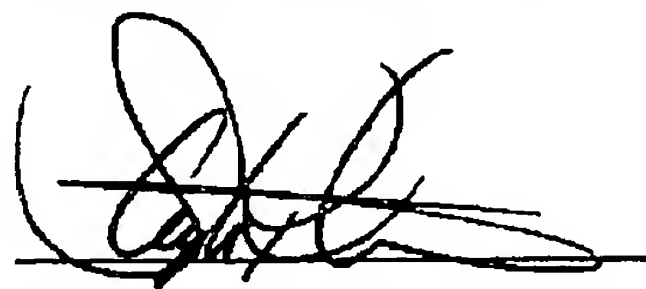
V. Allowable Subject Matter

Applicants thank the Examiner for the indication of allowable subject matter in claims 26 and 27. Applicants will defer placing claims 26 and 27 in independent form to a later action, if necessary.

VI. Conclusion

It is respectfully urged that the subject application is now in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

Respectfully submitted,



DATE: April 19, 2006

Stephen R. Tkacs
Reg. No. 46,430
WALDER INTELLECTUAL PROPERTY LAW, P.C.
P.O. Box 832745
Richardson, TX 75083
(214) 722-6422
AGENT FOR APPLICANTS